TARDEC

---TECHNICAL REPORT---

THE NATION'S LABORATORY FOR ADVANCED AUTOMOTIVE TECHNOLOGY

No. 13815

WINNER OF THE 1994 FEDERAL QUALITY IMPROVEMENT PROTOTYPE AWARD

Defining Paths for Driver Models with Hermite Parametric Curves

By W. Bylsma

Approved for public release; distribution is unlimited.

U.S. Army Tank-Automotive Research,
Development, and Engineering Center
Detroit Arsenal
Warren, Michigan 48397-5000

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	JUNE 2002	MAY - JUNE 2002

4. TITLE AND SUBTITLE
DEFINING PATHS FOR DRIVER MODELS WITH HERMITE PARAMETRIC CURVES

5. FUNDING NUMBERS

6. AUTHOR(S)
Wesley Bylsma

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Tank-automotive and
Armaments Command/National
Automotive Center
ATTN: AMSTA-TR-N/MS157
Warren, MI 48397-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

13815

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release: Distribution is unlimited

12b. DISTRIBUTION CODE

A

13. ABSTRACT *(Maximum 200 Words)*

A method (MATLAB function) to define continuous smooth and nonsmooth paths using Hermite parametric curves is presented. The defined path can be used as a reference signal in a driver model for feedback control of vehicle position. Four input formats are allowed that encompass two and three dimensional paths. The formulation can account for path crossover and provides a method to deal with the nonlinear relationship relating path distance and path parameter values.

14. SUBJECT TERMS
path creation, driver model, hermite parametric curves

15. NUMBER OF PAGES
15

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unclassified

Table of Contents

Technical Report 13815
June 2002

Defining Paths for Driver Models with Hermite Parametric Curves

W. Bylsma
U.S. Army Tank-automotive and Armaments Command
Research, Development and Engineering Center
National Automotive Center
Warren, Michigan 48397-5000

ABSTRACT
A method (MATLAB function) to define continuous smooth and nonsmooth paths using Hermite parametric curves is presented. The defined path can be used as a reference signal in a driver model for feedback control of vehicle position. Four input formats are allowed that encompass two and three dimensional paths. The formulation can account for path crossover and provides a method to deal with the nonlinear relationship relating path distance and path parameter values.

INTRODUCTION
In determining ground vehicle dynamic performance it is generally accepted that comparisons between simulated and test results will be compared for precision and accuracy. The test environment will define a particular course(s) that will be used to define the excitation to the vehicle. In the simulation environment the same course(s) must be replicated to excite the dynamics of the model. Each course used may, and for usefulness will, have different characteristics that excite the dynamics in different ways to obtain a complete spectrum of inputs to the system. The broader the spectrum the better characterization of the system. In the simulation environment driver models are developed to ensure that the vehicle model is traversing the desired course(s) in the same manner as in the test environment for vehicle parameters (speed, steering, etc.).

A path must be defined for each course that represents the desired vehicle trajectory and will be used as a reference signal in the driver model for feedback control to maintain the vehicle parameters (speed, steering, etc.). Ideally, each position on the path should be uniquely identifiable.

To satisfy these desired conditions each position on the path is defined by a point. A parametric curve is then defined through these points which will allow one parameter to define a unique position on the path. This formulation also provides for path crossover.

The number of points required and the method of interpolation between them to accurately represent the trajectory are issues that depend on the resolution required.

PATH DEFINITION
Hermite Cubic (similar to Catmull-Rom curves but tangents are specified and not inferred from neighboring points) are used here to represent the path because 1) they can be parameterized, 2) they have the desired property of passing directly through each point interpolating between, 3) they can be made to be continuous at each point, and 4) the tangent vectors can be explicitly specified.

Each spatial coordinate is specified as (y and z are the same)

$$x(t) = at^3 + bt^2 + ct + d \tag{1}$$

a third order polynomial (cubic) whose derivative is

$$\dot{x}(t) = 3at^2 + 2bt + c \tag{2}$$

where the parameter

$$t \in [0,1]. \tag{3}$$

The solution for the four coefficients requires four equations. Equations (1) and (2) are evaluated at both limits of (3) to get the solution for the coefficients in (1) as

$$\begin{bmatrix} x(0) \\ x(1) \\ \dot{x}(0) \\ \dot{x}(1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}. \tag{4}$$

Taking the inverse (see Appendix A) gives

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \dot{x}(0) \\ \dot{x}(1) \end{bmatrix} \tag{5}$$

The coefficients for each segment (between points) is defined by equations (1) and (3).

Since the parameter gives a unique position on the path it is important to relate this to the spatial domain in terms of knowing how far the vehicle has traveled on the path or what value of the parameter is needed to get to a certain distance on the path. Any point on a segment of the path is

$$r = r(t) = (x(t), y(t), z(t)), a \le t \le b \tag{6}$$

where the coefficients are different for each segment, but the limits in (6) on the parameter may be the same. From this the distance is related to the integral of the path speed

$$\dot{r}(t) = (\dot{x}(t), \dot{y}(t), \dot{z}(t)) \tag{7}$$

$$ds = |d\dot{r}| = |\dot{r}(t)|dt = vdt \tag{8}$$

$$s = \int_a^b |\dot{r}(t)|dt = \int_a^b \sqrt{\dot{x}^2(t) + \dot{y}^2(t) + \dot{z}^2(t)}dt \tag{9}$$

where (9) is over all path segments. The relationship in (9) is clearly nonlinear. A value of the parameter can easily be related to the path distance, but the path distance cannot easily be related to the parameter. However, since the parameter and distance along the path are both monotonically increasing values, a lookup table can be generated that can be used to lookup a value of the parameter for a particular path distance. With this solution the nonlinear relationship between path distance and parameter can easily be handled.

IMPLEMENTATION

Creation of a path requires points defining positions on the path and tangent vectors at each point to define the orientation of the path passing through that point.

Consideration is also given to the ability to create smooth or non-smooth ("sharp") paths. This is accomplished by allowing two tangent vectors to be specified for each point. The first is used if the point is at the end of a segment, the second if at the beginning of a segment. This allows "reorientation" at that point for the next segment of the path.

Consideration is also given for three dimensional as well as two dimensional paths. The three dimensional capability will allow better control for traversing a path with different elevations.

In the MATLAB implementation, a path is "created" by calling the function

```
makepath('input.path', 'output.coef',
         'output.tbl', dp, ds);
```

where "input.path" is the path definition file name, "output.coef" is the coefficient output file name, "output.tbl" is the parameter versus path distance file name, dp is the parameter increment used to generate the internal p and s relationship, and ds is the distance increment used to generate parameter versus path distance file "output.tbl" for equally spaced points on the path.

PATH FILE FORMATS

All files, input and output, are ASCII text files. The path input file, "input.path", is a sequence of points and tangent vectors defining the path. Any line that starts with a "%" or "#" or "*" is a comment. There are four possible input formats:

2D

Each line contains three values

[x y theta]

where the tangent vector is assumed to be

x'=cos(theta), y'=sin(theta)

Note that for the 2D format the magnitude of the tangent vector is assumed to be unity.

2D BRK

Each line contains four values

[x y theta theta1]

where the tangent vectors are assumed to be

x'=cos(theta), y'=sin(theta)

if at the end of a segment and

x'=cos(theta1), y'=sin(theta1)

if at the beginning of a segment.

3D

Each line contains six values

[x y z x' y' z']

where the tangent vector is

x' y' z'

3D BRK

Each line contains nine values

[x y z x' y' z' x'' y'' z'']

where the tangent vectors are assumed to be

x' y' z'

if at the end of a segment and

x'' y'' z''

if at the beginning of a segment.

The "output.coef" file has the form of

[0; ax; bx; cx; dx; ay; by; cy; dy; az; bz; cz; dz]
[1; ax; bx; cx; dx; ay; by; cy; dy; az; bz; cz; dz]
[2; ax; bx; cx; dx; ay; by; cy; dy; az; bz; cz; dz]
...

where the first line is the point number then three sets of four lines for x, y, and z coefficient values respectively.

The "output.tbl" file has the form of

[p, s]

RESULTS

The following will illustrate the usage for smooth and nonsmooth paths.

The smooth path example is one typical of a lane change for a vehicle. The path is described as follows

```
---BOF---
%lane
%3D format but 2D curve
0  0  0  5  0  0
5  0  0  5  0  0
10  5  0  5  1  0
15  5  0  5  -1  0
20  0  0  5  0  0
25  0  0  5  0  0
---EOF---
```

See Appendix B for the "output.coef" and "output.tbl" files. Figure 1 shows the continuous path generated from the input points above.

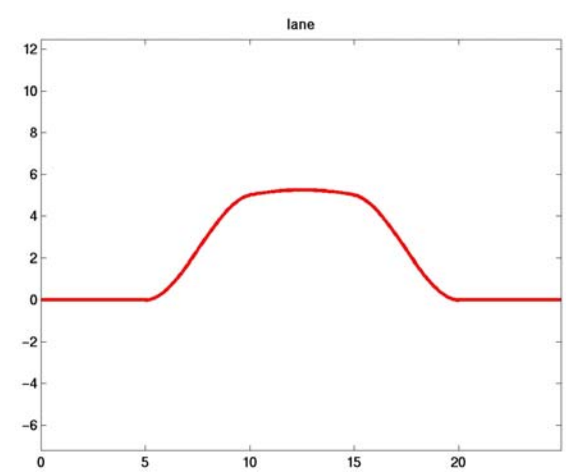

Figure 1 - Smooth Path

Figure 2 shows the nonlinear relationship between the path distance and parameter.

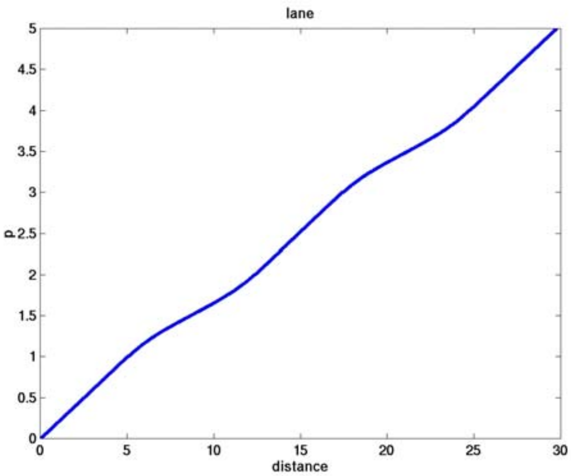

Figure 2 - Distance versus Parameter (S)

The nonsmooth path example is a three sided square and line segment that demonstrates path crossover. The path is described as follows

```
---BOF---
%square with crossover
%2D format
0 0 90
0 5 90 0
5 5 0 -90
5 0 -90 135
-10 10 135
---EOF---
```

See Appendix C for the "output.coef" and "output.tbl" files. Figure 3 shows the continuous path generated from the input points above for the nonsmooth path.

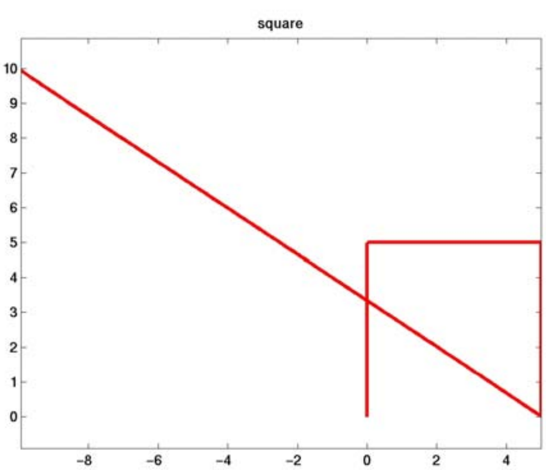

Figure 3 - Nonsmooth Path

Figure 4 shows the nonlinear relationship between the path distance and parameter for the nonsmooth crossover path.

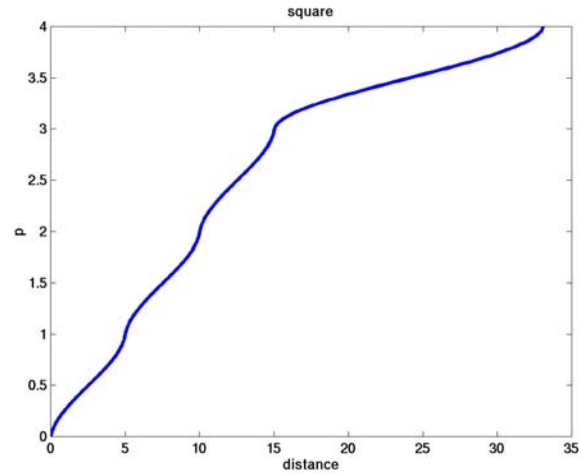

Figure 4 - Distance versus Parameter (NS)

CONCLUSION

The results demonstrate the utility of Hermite parametric curves for input to driver models and their versatility by being able to describe smooth/nonsmooth and multiple crossover paths with ease.

Due to the monotonic nature of the path parameter and distance a lookup table can be used to relate the two path parameters as shown in Figures 2 and 4. For the lane and square path examples the end distance and parameter values are 29.7 and 4.98, 33.0 and 3.96 respectively (dp = 0.01, ds= 0.1). These values correlate with the number and displacement of the points in the respective example input file.

CONTACT

The author is a an engineer at the U.S. Army Tank-automotive and Armaments Command, Research, Development and Engineering Center (TACOM-TARDEC). Interested parties can contact the author at the U.S. Army Tank-automotive and Armaments Command, ATTN: AMSTA-TR-N/MS157, Warren, Michigan 48397-5000, bylsmaw@tacom.army.mil.

REFERENCES

Fundamentals of Interactive Computer Graphics, J.D. Foley and A. Van Dam. Addison-Wesley, 1984. ISBN 0-201-14468-9

DEFINITIONS, ACRONYMS, ABBREVIATIONS

S - Smooth, NS - Nonsmooth, TACOM - U.S. Army Tank-automotive and Armaments Command, TARDEC - TACOM Research, Development and Engineering Center, NAC - National Automotive Center.

APPENDIX A - Coefficient Matrix Inverse

Each point is numbered, starting from zero, so the parameter varies from 0 to 1, starting at p_n to p_{n+1} where p_n of the next point will equal p_{n+1} of the previous point. For each interval between points then

$$x(p-p_n)=a(p-p_n)^3+b(p-p_n)^2+c(p-p_n)+d$$

and

$$\dot{x}(p-p_n)=3a(p-p_n)^2+2b(p-p_n)+c$$

where these two equations are evaluated at

$$x(p-p_n)\big|_{p=p_n}$$
$$x(p-p_n)\big|_{p=p_{n+1}}$$
$$\dot{x}(p-p_n)\big|_{p=p_n}$$
$$\dot{x}(p-p_n)\big|_{p=p_{n+1}}$$

gives

$$\begin{bmatrix} x(p-p_n)\big|_{p=p_n} \\ x(p-p_n)\big|_{p=p_{n+1}} \\ \dot{x}(p-p_n)\big|_{p=p_n} \\ \dot{x}(p-p_n)\big|_{p=p_{n+1}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

Now

$$[A^{-1}]_{ij}=\frac{1}{\det A}Cofactor(A^T{}_{ij})$$

where

$$Cofactor(A_{ij})=(-1)^{i+j}D_{ij}$$

and D_{ij} =subdeterminant is formed by deleting the ith row and the jth column of D_{ij} and taking the determinant.

$$\det A=0(-1)^{1+1}D_{11}+0(-1)^{1+2}D_{12}+0(-1)^{1+3}D_{13}+1(-1)^{1+4}\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 3 & 2 & 1 \end{vmatrix}=(-1)^{1+4}(-1)^{2+3}\begin{vmatrix} 1 & 1 \\ 3 & 2 \end{vmatrix}=(-1)(-1)(2-3)=-1$$

$$C_{11} = (-1)^{1+1}\begin{vmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 2 & 1 & 0 \end{vmatrix} = (-1)^{1+1}(1)(-1)^{2+2}\begin{vmatrix} 1 & 1 \\ 2 & 0 \end{vmatrix} = (1)(1)(1)(0-2) = -2$$

$$C_{12} = (-1)^{1+2}\begin{vmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 3 & 1 & 0 \end{vmatrix} = (-1)^{1+2}(1)(-1)^{1+3}\begin{vmatrix} 0 & 1 \\ 3 & 1 \end{vmatrix} = (-1)(1)(1)(0-3) = 3$$

$$C_{13} = (-1)^{1+3}\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 3 & 2 & 0 \end{vmatrix} = (-1)^{1+3}(1)(-1)^{1+3}\begin{vmatrix} 0 & 0 \\ 3 & 2 \end{vmatrix} = (1)(1)(1)(0) = 0$$

$$C_{14} = (-1)^{1+4}\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 3 & 2 & 1 \end{vmatrix} = (-1)^{1+4}(1)(-1)^{2+3}\begin{vmatrix} 1 & 1 \\ 3 & 2 \end{vmatrix} = (-1)(1)(-1)(2-3) = -1$$

$$C_{21} = (-1)^{2+1}\begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 2 & 1 & 0 \end{vmatrix} = (-1)^{2+1}(1)(-1)^{1+3}\begin{vmatrix} 0 & 1 \\ 2 & 1 \end{vmatrix} = (-1)(1)(1)(0-2) = 2$$

$$C_{22} = (-1)^{2+2}\begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 3 & 1 & 0 \end{vmatrix} = (-1)^{2+2}(1)(-1)^{1+3}\begin{vmatrix} 0 & 1 \\ 3 & 1 \end{vmatrix} = (1)(1)(1)(0-3) = -3$$

$$C_{23} = (-1)^{2+3}\begin{vmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 3 & 2 & 0 \end{vmatrix} = (-1)^{2+3}(1)(-1)^{1+3}\begin{vmatrix} 0 & 0 \\ 3 & 2 \end{vmatrix} = (-1)(1)(1)(0-0) = 0$$

$$C_{24} = (-1)^{2+4}\begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 3 & 2 & 1 \end{vmatrix} = (-1)^{2+4}(0) = 0$$

$$C_{31} = (-1)^{3+1}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 2 & 1 & 0 \end{vmatrix} = (-1)^{3+1}(1)(-1)^{1+3}\begin{vmatrix} 1 & 1 \\ 2 & 1 \end{vmatrix} = (1)(1)(1)(1-2) = -1$$

$$C_{32} = (-1)^{3+2}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 3 & 1 & 0 \end{vmatrix} = (-1)^{3+2}(1)(-1)^{1+3}\begin{vmatrix} 1 & 1 \\ 3 & 1 \end{vmatrix} = (-1)(1)(1)(1-3) = 2$$

$$C_{33} = (-1)^{3+3}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 3 & 2 & 0 \end{vmatrix} = (-1)^{3+3}(1)(-1)^{1+3}\begin{vmatrix} 1 & 1 \\ 3 & 2 \end{vmatrix} = (1)(1)(1)(2-3) = -1$$

$$C_{34} = (-1)^{3+4}\begin{vmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 3 & 2 & 1 \end{vmatrix} = (-1)^{3+4}(0) = 0$$

$$C_{41} = (-1)^{4+1}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{vmatrix} = (-1)^{4+1}(1)(-1)^{1+3}\begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} = (-1)(1)(1)(1-0) = -1$$

$$C_{42} = (-1)^{4+2}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{vmatrix} = (-1)^{4+2}(1)(-1)^{1+3}\begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} = (1)(1)(1)(1-0) = 1$$

$$C_{43} = (-1)^{4+3}\begin{vmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{vmatrix} = (-1)^{4+3}(0) = 0$$

$$C_{44} = (-1)^{4+4}\begin{vmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{vmatrix} = (-1)^{4+4}(0) = 0$$

$$[A^{-1}]_{ij} = \frac{1}{-1}\begin{bmatrix} -2 & 3 & 0 & -1 \\ 2 & -3 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & 1 & 0 & 0 \end{bmatrix}^T = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

APPENDIX B - Lane Path Example Output Files

```
---BOF---"output.coef"
0.000000
0.000000
0.000000
5.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
1.000000
0.000000
0.000000
5.000000
5.000000
-9.000000
14.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
2.000000
0.000000
0.000000
5.000000
10.000000
0.000000
-1.000000
1.000000
5.000000
0.000000
0.000000
0.000000
0.000000
3.000000
0.000000
0.000000
5.000000
15.000000
9.000000
-13.000000
-1.000000
5.000000
0.000000
0.000000
0.000000
0.000000
4.000000
0.000000
0.000000
5.000000
20.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000
---EOF---"output.coef"
```

```
---BOF---"output.tbl"
0.000000 0.000000
0.100000 0.010000
0.200000 0.030000
0.300000 0.050000
0.400000 0.070000
0.500000 0.090000
0.600000 0.110000
0.700000 0.130000
0.800000 0.150000
0.900000 0.170000
1.000000 0.190000
1.100000 0.210000
1.200000 0.230000
1.300000 0.250000
1.400000 0.270000
1.500000 0.290000
1.600000 0.310000
1.700000 0.330000
1.800000 0.350000
1.900000 0.370000
2.000000 0.390000
2.100000 0.410000
2.200000 0.430000
2.300000 0.450000
2.400000 0.470000
2.500000 0.490000
2.600000 0.510000
2.700000 0.530000
2.800000 0.550000
2.900000 0.570000
3.000000 0.590000
3.100000 0.610000
3.200000 0.630000
3.300000 0.650000
3.400000 0.670000
3.500000 0.690000
3.600000 0.710000
3.700000 0.730000
3.800000 0.750000
3.900000 0.770000
4.000000 0.790000
4.100000 0.810000
4.200000 0.830000
4.300000 0.850000
4.400000 0.870000
4.500000 0.890000
4.600000 0.910000
4.700000 0.930000
4.800000 0.950000
4.900000 0.970000
5.000000 0.990000
5.100000 1.000000
5.200000 1.019925
5.300000 1.039573
5.400000 1.058784
5.500000 1.077465
5.600000 1.095575
5.700000 1.113117
5.800000 1.130114
5.900000 1.146542
6.000000 1.162500
6.100000 1.178001
6.200000 1.193079
6.300000 1.207782
6.400000 1.222136
6.500000 1.236166
6.600000 1.249930
6.700000 1.263398
6.800000 1.276642
6.900000 1.289679
7.000000 1.302496
7.100000 1.315137
```

```
7.200000 1.327620
7.300000 1.339957
7.400000 1.352142
7.500000 1.364208
7.600000 1.376166
7.700000 1.388025
7.800000 1.399795
7.900000 1.411478
8.000000 1.423091
8.100000 1.434644
8.200000 1.446144
8.300000 1.457600
8.400000 1.469019
8.500000 1.480408
8.600000 1.491774
8.700000 1.503126
8.800000 1.514473
8.900000 1.525819
9.000000 1.537174
9.100000 1.548543
9.200000 1.559935
9.300000 1.571360
9.400000 1.582823
9.500000 1.594332
9.600000 1.605896
9.700000 1.617521
9.800000 1.629216
9.900000 1.640998
10.000000 1.652874
10.100000 1.664851
10.200000 1.676938
10.300000 1.689147
10.400000 1.701502
10.500000 1.714014
10.600000 1.726687
10.700000 1.739535
10.800000 1.752605
10.900000 1.765891
11.000000 1.779404
11.100000 1.793209
11.200000 1.807295
11.300000 1.821699
11.400000 1.836469
11.500000 1.851610
11.600000 1.867189
11.700000 1.883225
11.800000 1.899742
11.900000 1.916831
12.000000 1.934468
12.100000 1.952676
12.200000 1.971451
12.300000 1.990744
12.400000 2.000355
12.500000 2.019990
12.600000 2.039652
12.700000 2.059342
12.800000 2.079058
12.900000 2.098799
13.000000 2.118565
13.100000 2.138353
13.200000 2.158164
13.300000 2.177995
13.400000 2.197845
13.500000 2.217715
13.600000 2.237601
13.700000 2.257504
13.800000 2.277422
13.900000 2.297353
14.000000 2.317297
14.100000 2.337252
14.200000 2.357217
14.300000 2.377191
14.400000 2.397172
```

```
14.500000 2.417160
14.600000 2.437152
14.700000 2.457148
14.800000 2.477147
14.900000 2.497147
15.000000 2.517147
15.100000 2.537145
15.200000 2.557141
15.300000 2.577132
15.400000 2.597119
15.500000 2.617098
15.600000 2.637070
15.700000 2.657033
15.800000 2.676986
15.900000 2.696927
16.000000 2.716856
16.100000 2.736770
16.200000 2.756670
16.300000 2.776553
16.400000 2.796418
16.500000 2.816265
16.600000 2.836091
16.700000 2.855897
16.800000 2.875680
16.900000 2.895441
17.000000 2.915177
17.100000 2.934887
17.200000 2.954571
17.300000 2.974227
17.400000 2.993855
17.500000 3.003429
17.600000 3.022622
17.700000 3.041290
17.800000 3.059384
17.900000 3.076882
18.000000 3.093832
18.100000 3.110276
18.200000 3.126189
18.300000 3.141677
18.400000 3.156733
18.500000 3.171425
18.600000 3.185752
18.700000 3.199791
18.800000 3.213510
18.900000 3.226983
19.000000 3.240228
19.100000 3.253232
19.200000 3.266048
19.300000 3.278691
19.400000 3.291164
19.500000 3.303480
19.600000 3.315666
19.700000 3.327732
19.800000 3.339690
19.900000 3.351538
20.000000 3.363298
20.100000 3.374979
20.200000 3.386592
20.300000 3.398145
20.400000 3.409644
20.500000 3.421096
20.600000 3.432511
20.700000 3.443898
20.800000 3.455265
20.900000 3.466617
21.000000 3.477963
21.100000 3.489310
21.200000 3.500665
21.300000 3.512036
21.400000 3.523431
21.500000 3.534857
21.600000 3.546321
21.700000 3.557830
```

21.800000	3.569394
21.900000	3.581024
22.000000	3.592729
22.100000	3.604516
22.200000	3.616392
22.300000	3.628369
22.400000	3.640459
22.500000	3.652687
22.600000	3.665049
22.700000	3.677560
22.800000	3.690233
22.900000	3.703111
23.000000	3.716183
23.100000	3.729464
23.200000	3.743013
23.300000	3.756821
23.400000	3.770911
23.500000	3.785350
23.600000	3.800109
23.700000	3.815298
23.800000	3.830876
23.900000	3.846946
24.000000	3.863506
24.100000	3.880586
24.200000	3.898256
24.300000	3.916503
24.400000	3.935309
24.500000	3.954625
24.600000	3.974343
24.700000	3.994294
24.800000	4.004294
24.900000	4.024294
25.000000	4.044294
25.100000	4.064294
25.200000	4.084294
25.300000	4.104294
25.400000	4.124294
25.500000	4.144294
25.600000	4.164294
25.700000	4.184294
25.800000	4.204294
25.900000	4.224294
26.000000	4.244294
26.100000	4.264294
26.200000	4.284294
26.300000	4.304294
26.400000	4.324294
26.500000	4.344294
26.600000	4.364294
26.700000	4.384294
26.800000	4.404294
26.900000	4.424294
27.000000	4.444294
27.100000	4.464294
27.200000	4.484294
27.300000	4.504294
27.400000	4.524294
27.500000	4.544294
27.600000	4.564294
27.700000	4.584294
27.800000	4.604294
27.900000	4.624294
28.000000	4.644294
28.100000	4.664294
28.200000	4.684294
28.300000	4.704294
28.400000	4.724294
28.500000	4.744294
28.600000	4.764294
28.700000	4.784294
28.800000	4.804294
28.900000	4.824294
29.000000	4.844294
29.100000	4.864294
29.200000	4.884294
29.300000	4.904294
29.400000	4.924294
29.500000	4.944294
29.600000	4.964294
29.700000	4.984294

---EOF---"output.tbl"

APPENDIX C - Square Path Example Output Files

```
---BOF---"output.coef"
0.000000
0.000000
0.000000
0.000000
0.000000
-8.000000
12.000000
1.000000
0.000000
0.000000
0.000000
0.000000
0.000000
1.000000
-8.000000
12.000000
1.000000
0.000000
0.000000
0.000000
0.000000
5.000000
0.000000
0.000000
0.000000
0.000000
2.000000
0.000000
0.000000
0.000000
5.000000
8.000000
-12.000000
-1.000000
5.000000
0.000000
0.000000
0.000000
0.000000
3.000000
28.585786
-42.878680
-0.707107
5.000000
-18.585786
27.878680
0.707107
0.000000
0.000000
0.000000
0.000000
0.000000
---EOF---"output.coef"
```

```
---BOF---"output.tbl"
0.000000 0.000000
0.100000 0.052260
0.200000 0.089609
0.300000 0.119547
0.400000 0.145507
0.500000 0.169006
0.600000 0.190678
0.700000 0.210989
0.800000 0.230256
0.900000 0.248655
1.000000 0.266356
1.100000 0.283490
1.200000 0.300150
1.300000 0.316362
1.400000 0.332239
1.500000 0.347808
1.600000 0.363115
1.700000 0.378205
1.800000 0.393101
1.900000 0.407841
2.000000 0.422447
2.100000 0.436945
2.200000 0.451362
2.300000 0.465712
2.400000 0.480025
2.500000 0.494314
2.600000 0.508603
2.700000 0.522912
2.800000 0.537259
2.900000 0.551666
3.000000 0.566158
3.100000 0.580749
3.200000 0.595479
3.300000 0.610352
3.400000 0.625429
3.500000 0.640706
3.600000 0.656258
3.700000 0.672096
3.800000 0.688285
3.900000 0.704894
4.000000 0.721968
4.100000 0.739608
4.200000 0.757959
4.300000 0.777135
4.400000 0.797339
4.500000 0.818846
4.600000 0.842128
4.700000 0.867846
4.800000 0.897329
4.900000 0.933661
5.000000 0.990400
5.100000 1.048000
5.200000 1.086372
5.300000 1.116831
5.400000 1.143143
5.500000 1.166817
5.600000 1.188653
5.700000 1.209089
5.800000 1.228434
5.900000 1.246909
6.000000 1.264681
6.100000 1.281874
6.200000 1.298563
6.300000 1.314819
6.400000 1.330735
6.500000 1.346322
6.600000 1.361660
6.700000 1.376762
6.800000 1.391680
6.900000 1.406430
7.000000 1.421052
7.100000 1.435557
7.200000 1.449983
7.300000 1.464337
7.400000 1.478652
7.500000 1.492943
7.600000 1.507231
7.700000 1.521536
7.800000 1.535880
7.900000 1.550278
8.000000 1.564763
8.100000 1.579342
8.200000 1.594059
8.300000 1.608919
8.400000 1.623973
8.500000 1.639228
8.600000 1.654754
8.700000 1.670553
8.800000 1.686720
8.900000 1.703279
9.000000 1.720293
9.100000 1.737899
9.200000 1.756174
9.300000 1.775259
9.400000 1.795355
9.500000 1.816733
9.600000 1.839773
9.700000 1.865262
9.800000 1.894291
9.900000 1.929616
10.000000 1.982566
10.100000 2.043514
10.200000 2.083135
10.300000 2.114115
10.400000 2.140778
10.500000 2.164629
10.600000 2.186608
10.700000 2.207162
10.800000 2.226605
10.900000 2.245164
11.000000 2.263006
11.100000 2.280259
11.200000 2.296974
11.300000 2.313276
11.400000 2.329222
11.500000 2.344836
11.600000 2.360204
11.700000 2.375320
11.800000 2.390260
11.900000 2.405019
12.000000 2.419655
12.100000 2.434168
12.200000 2.448599
12.300000 2.462961
12.400000 2.477279
12.500000 2.491571
12.600000 2.505859
12.700000 2.520160
12.800000 2.534501
12.900000 2.548893
13.000000 2.563369
13.100000 2.577940
13.200000 2.592639
13.300000 2.607488
13.400000 2.622517
13.500000 2.637758
13.600000 2.653251
13.700000 2.669023
13.800000 2.685155
13.900000 2.701663
14.000000 2.718643
14.100000 2.736191
14.200000 2.754389
14.300000 2.773384
14.400000 2.793372
14.500000 2.814620
14.600000 2.837501
14.700000 2.862677
14.800000 2.891253
14.900000 2.925869
15.000000 2.975631
15.100000 3.022848
15.200000 3.043864
15.300000 3.059505
15.400000 3.072380
15.500000 3.083757
15.600000 3.094082
15.700000 3.103613
15.800000 3.112527
15.900000 3.120953
16.000000 3.128921
16.100000 3.136503
16.200000 3.143816
16.300000 3.150903
16.400000 3.157693
16.500000 3.164283
16.600000 3.170740
16.700000 3.176959
16.800000 3.183057
16.900000 3.189037
17.000000 3.194837
17.100000 3.200581
17.200000 3.206153
17.300000 3.211670
17.400000 3.217066
17.500000 3.222389
17.600000 3.227626
17.700000 3.232784
17.800000 3.237876
17.900000 3.242892
18.000000 3.247853
18.100000 3.252745
18.200000 3.257585
18.300000 3.262371
18.400000 3.267101
18.500000 3.271792
18.600000 3.276421
18.700000 3.281030
18.800000 3.285567
18.900000 3.290102
19.000000 3.294554
19.100000 3.299007
19.200000 3.303399
19.300000 3.307773
19.400000 3.312113
19.500000 3.316416
19.600000 3.320709
19.700000 3.324947
19.800000 3.329185
19.900000 3.333375
20.000000 3.337553
20.100000 3.341709
20.200000 3.345833
20.300000 3.349956
20.400000 3.354031
20.500000 3.358105
20.600000 3.362154
20.700000 3.366183
20.800000 3.370209
20.900000 3.374197
21.000000 3.378184
21.100000 3.382151
21.200000 3.386102
21.300000 3.390052
21.400000 3.393969
21.500000 3.397886
21.600000 3.401790
21.700000 3.405677
21.800000 3.409565
21.900000 3.413429
22.000000 3.417291
```

22.100000	3.421145
22.200000	3.424984
22.300000	3.428823
22.400000	3.432648
22.500000	3.436467
22.600000	3.440285
22.700000	3.444088
22.800000	3.447891
22.900000	3.451688
23.000000	3.455478
23.100000	3.459267
23.200000	3.463049
23.300000	3.466828
23.400000	3.470606
23.500000	3.474378
23.600000	3.478150
23.700000	3.481920
23.800000	3.485688
23.900000	3.489456
24.000000	3.493222
24.100000	3.496988
24.200000	3.500755
24.300000	3.504523
24.400000	3.508290
24.500000	3.512060
24.600000	3.515832
24.700000	3.519604
24.800000	3.523383
24.900000	3.527162
25.000000	3.530944
25.100000	3.534734
25.200000	3.538524
25.300000	3.542321
25.400000	3.546124
25.500000	3.549927
25.600000	3.553746
25.700000	3.557565
25.800000	3.561392
25.900000	3.565230
26.000000	3.569069
26.100000	3.572925
26.200000	3.576786
26.300000	3.580652
26.400000	3.584540
26.500000	3.588427
26.600000	3.592333
26.700000	3.596250
26.800000	3.600168
26.900000	3.604119
27.000000	3.608069
27.100000	3.612039
27.200000	3.616026
27.300000	3.620014
27.400000	3.624042
27.500000	3.628071
27.600000	3.632123
27.700000	3.636197
27.800000	3.640274
27.900000	3.644398
28.000000	3.648521
28.100000	3.652680
28.200000	3.656858
28.300000	3.661051
28.400000	3.665290
28.500000	3.669528
28.600000	3.673824
28.700000	3.678127
28.800000	3.682471
28.900000	3.686846
29.000000	3.691242
29.100000	3.695694
29.200000	3.700149
29.300000	3.704686
29.400000	3.709223
29.500000	3.713837
29.600000	3.718466
29.700000	3.723163
29.800000	3.727894
29.900000	3.732685
30.000000	3.737525
30.100000	3.742424
30.200000	3.747385
30.300000	3.752408
30.400000	3.757500
30.500000	3.762666
30.600000	3.767903
30.700000	3.773236
30.800000	3.778632
30.900000	3.784158
31.000000	3.789730
31.100000	3.795485
31.200000	3.801297
31.300000	3.807277
31.400000	3.813388
31.500000	3.819608
31.600000	3.826079
31.700000	3.832687
31.800000	3.839477
31.900000	3.846582
32.000000	3.853917
32.100000	3.861523
32.200000	3.869492
32.300000	3.877947
32.400000	3.886895
32.500000	3.896466
32.600000	3.906839
32.700000	3.918276
32.800000	3.931322
32.900000	3.947092
33.000000	3.968562

---EOF---"output.tbl"

APPENDIX D - Makepath Listing

```
% Make Path (makepath.m)
%
% Usage:
%
% makepath [input] [output.coef] [output.tbl] [dp] [ds]
%
% Example: makepath('input.path', 'output.coef', 'output.tbl', 0.01, 0.1);
%
% where input format is
%
% 2D = [x y theta] where x'=cos(theta), y'=sin(theta)
% 2D BRK = [x y theta theta1]
% 3D = [x y z x' y' z']
% 3D BRK = [x y z x' y' z' x' y' z']
%
% for each point.
%
% If 2nd set of tangents is encountered it will restart at the same
% point with the new tangent vector.
%
% Comment lines start with '%' or '#' or '*'
%
% dp = parameter increment, ds = distance increment
%
function [A]=makepath(in,out,tbl,dp,ds);

LINE_LEN= 1024; % line length
MAX_PTS= 1000; % max number of points
PI= 4.0*atan(1.0);
DEG2RAD= PI/180.0;
FLG_2D =3;
FLG_2D_BRK= 4;
FLG_3D= 6;
FLG_3D_BRK= 9;
mh = [2.0, -2.0, 1.0, 1.0; -3.0, 3.0, -2.0, -1.0; 0.0, 0.0, 1.0, 0.0; 1.0, 0.0, 0.0, 0.0]; % transform matrix

fprintf('\n\n\nMake Path\n\n');

%---open files and process
fin = fopen(in,'rt');
fout = fopen(out,'w');
fout1 = fopen(tbl,'w');

fprintf('Processing %s ...\n',in);

num_pts = 1;
val = 0;

line = fgetl(fin);
while (line(1) == '%' | line(1) == '#' | line(1) == '*'),
  line = fgetl(fin);
end

while (line ~= -1),
  fprintf('[%s]\n',line);
  val = sscanf(line,'%f %f %f %f %f %f %f %f %f'); % &x, &y, &z, &xp, &yp, &zp, &xpn, &ypn, &zpn);
  lval = length(val);
  val = [val',[zeros(9-length(val),1)']]';
  x = val(1);
  y = val(2);
  z = val(3);
  xp = val(4);
  yp = val(5);
  zp = val(6);
  xpn = val(7);
  ypn = val(8);
  zpn = val(9);
  if (val == -1)
    break;
  end
  switch (lval)
    case {FLG_2D}
      %---2D = [x y theta] where x'=cos(theta), y'=sin(theta)
      pts.flg(num_pts) = FLG_2D;
```

```
      pts.x(num_pts) = x;
      pts.y(num_pts) = y;
      pts.z(num_pts) = 0.0;
      pts.zp(num_pts) = z;

      pts.xp(num_pts) = cos(DEG2RAD*pts.zp(num_pts));
      pts.yp(num_pts) = sin(DEG2RAD*pts.zp(num_pts));
      pts.zp(num_pts) = 0.0;

      pts.xpn(num_pts) = 0.0;
      pts.ypn(num_pts) = 0.0;
      pts.zpn(num_pts) = 0.0;
%      fprintf('--%d-%f %f %f %f %f %f %f %f %f\n', pts.flg(num_pts),...
%                                                      pts.x(num_pts), pts.y(num_pts), pts.z(num_pts),...
%                                                      pts.xp(num_pts), pts.yp(num_pts), pts.zp(num_pts),...
%                                                      pts.xpn(num_pts), pts.ypn(num_pts), pts.zpn(num_pts));

    case {FLG_2D_BRK}
      %---2D BRK = [x y theta theta1]
      pts.flg(num_pts) = FLG_2D_BRK;
      pts.x(num_pts) = x;
      pts.y(num_pts) = y;
      pts.z(num_pts) = 0.0;
      pts.zp(num_pts) = z;
      pts.zpn(num_pts) = xp;

      pts.xp(num_pts) = cos(DEG2RAD*pts.zp(num_pts));
      pts.yp(num_pts) = sin(DEG2RAD*pts.zp(num_pts));
      pts.zp(num_pts) = 0.0;

      pts.xpn(num_pts) = cos(DEG2RAD*pts.zpn(num_pts));
      pts.ypn(num_pts) = sin(DEG2RAD*pts.zpn(num_pts));
      pts.zpn(num_pts) = 0.0;
%      fprintf('--%d-%f %f %f %f %f %f %f %f %f\n', pts.flg(num_pts),...
%                                                    pts.x(num_pts), pts.y(num_pts), pts.z(num_pts),...
%                                                    pts.xp(num_pts), pts.yp(num_pts), pts.zp(num_pts),...
%                                                    pts.xpn(num_pts), pts.ypn(num_pts), pts.zpn(num_pts));

    case {FLG_3D}
      %---3D = [x y z x' y' z']
      pts.flg(num_pts) = FLG_3D;
      pts.x(num_pts) = x;
      pts.y(num_pts) = y;
      pts.z(num_pts) = z;
      pts.xp(num_pts) = xp;
      pts.yp(num_pts) = yp;
      pts.zp(num_pts) = zp;
      pts.xpn(num_pts) = 0.0;
      pts.ypn(num_pts) = 0.0;
      pts.zpn(num_pts) = 0.0;

    case {FLG_3D_BRK}
      %---3D BRK = [x y z x' y' z' x' y' z']
      pts.flg(num_pts) = FLG_3D_BRK;
      pts.x(num_pts) = x;
      pts.y(num_pts) = y;
      pts.z(num_pts) = z;
      pts.xp(num_pts) = xp;
      pts.yp(num_pts) = yp;
      pts.zp(num_pts) = zp;
      pts.xpn(num_pts) = xpn;
      pts.ypn(num_pts) = ypn;
      pts.zpn(num_pts) = zpn;

    otherwise
      fprintf('Error: Undefined number of values for point [%d]\n',k);
      exit(1);
  end
%fprintf(fout,'%d-%f %f %f %f %f %f %f %f %f\n', pts.flg(num_pts),....
%                                                  pts.x(num_pts), pts.y(num_pts), pts.z(num_pts),...
%                                                  pts.xp(num_pts), pts.yp(num_pts), pts.zp(num_pts),....
%                                                  pts.xpn(num_pts), pts.ypn(num_pts), pts.zpn(num_pts));
  num_pts = num_pts + 1;
  if (num_pts >= MAX_PTS)
    fprintf('Error: MAX_PTS=%d exceeded\n',MAX_PTS);
    stop;
  end
```

```
  line = fgetl(fin);
  while (line(1) == '%' | line(1) == '#' | line(1) == '*'),
    line = fgetl(fin);
  end
end

fclose(fin);

fprintf('---Read in %d points\n',num_pts);

fprintf('Processing %s ...\n',out);

%---create coefficients
for i = 1 : num_pts - 2,
  %---set vector to get coefficients between points
  %---x
  inv.a = pts.x(i);
  inv.b = pts.x(i+1);
  if ((pts.flg(i) == FLG_2D_BRK) | (pts.flg(i) == FLG_3D_BRK))
    inv.c = pts.xpn(i);
  else
    inv.c = pts.xp(i);
  end
  inv.d = pts.xp(i+1);
  cx(i,:) = [mh*[inv.a;inv.b;inv.c;inv.d]]';

  %---y
  inv.a = pts.y(i);
  inv.b = pts.y(i+1);
  if ((pts.flg(i) == FLG_2D_BRK) | (pts.flg(i) == FLG_3D_BRK))
    inv.c = pts.ypn(i);
  else
    inv.c = pts.yp(i);
  end
  inv.d = pts.yp(i+1);
  cy(i,:) = [mh*[inv.a;inv.b;inv.c;inv.d]]';

  %---z
  inv.a = pts.z(i);
  inv.b = pts.z(i+1);
  if ((pts.flg(i) == FLG_2D_BRK) | (pts.flg(i) == FLG_3D_BRK))
    inv.c = pts.zpn(i);
  else
    inv.c = pts.zp(i);
  end
  inv.d = pts.zp(i+1);
  cz(i,:) = [mh*[inv.a;inv.b;inv.c;inv.d]]';

  %--- print out coefficients
  fprintf(fout,'%f\n',i-1);
  fprintf(fout,'%f\n%f\n%f\n%f\n',cx(i,1),cx(i,2),cx(i,3),cx(i,4));
  fprintf(fout,'%f\n%f\n%f\n%f\n',cy(i,1),cy(i,2),cy(i,3),cy(i,4));
  fprintf(fout,'%f\n%f\n%f\n%f\n',cz(i,1),cz(i,2),cz(i,3),cz(i,4));

end

fclose(fout) ;

%---create p,s table at same time
sump(1,1)=0.0; % start at 0.0
sump(1,2)=0.0;
cnt = 2;

for i = 1 : num_pts - 2,
  for j = 0 : dp : 1,
    ptx(cnt) = [j^3 j^2 j 1]*cx(i,:)';
    ptvx(cnt) = [3*j^2 2*j 1 0]*cx(i,:)';
    pty(cnt) = [j^3 j^2 j 1]*cy(i,:)';
    ptvy(cnt) = [3*j^2 2*j 1 0]*cy(i,:)';
    ptz(cnt) = [j^3 j^2 j 1]*cz(i,:)';
    ptvz(cnt) = [3*j^2 2*j 1 0]*cz(i,:)';

    sump(cnt,1) = (i-1)+j;
    sump(cnt,2) = sqrt( ptvx(cnt)^2 + ptvy(cnt)^2 + ptvz(cnt)^2 ) * dp;
    cnt = cnt + 1;
  end
end
```

```
%---make plots
figure(1);
clf;
cnt=1;

for j = 0 : ds : sum(sump(:,2)),
  p = interp1(cumsum(sump(:,2)),sump(:,1),j,'linear');
  fprintf(fout1,'%f %f\n',j,p);
  i = floor(p) + 1;
  p = mod(p,1);
  ptx(cnt) = [p^3 p^2 p 1]*cx(i,:)';
  ptvx(cnt) = [3*p^2 2*p 1 0]*cx(i,:)';
  pty(cnt) = [p^3 p^2 p 1]*cy(i,:)';
  ptvy(cnt) = [3*p^2 2*p 1 0]*cy(i,:)';
  ptz(cnt) = [p^3 p^2 p 1]*cz(i,:)';
  ptvz(cnt) = [3*p^2 2*p 1 0]*cz(i,:)';
  cnt = cnt + 1;
end
fclose(fout1);

h = plot(ptx(1:cnt-1),pty(1:cnt-1),'r');
% print('-djpeg99',[char(gg(i)) '.jpg'])
% print -dps2 -Past4000tn %see help print
set(gca,'FontSize',12);
set(gca,'FontWeight','bold');
set(h,'LineWidth',3);
axis equal

figure(2);
clf;

h = plot(cumsum(sump(:,2)),sump(:,1))
set(gca,'FontSize',12)
set(gca,'FontWeight','bold');
set(h,'LineWidth',3);
xlabel('distance');
ylabel('p');

figure(3);
clf;
h=plot(sump(:,1),sump(:,2))
set(gca,'FontSize',12)
set(gca,'FontWeight','bold');
set(h,'LineWidth',3);
xlabel('p');
ylabel('vel');

%end
```